

# Refactoring, reengineering and evolution: paths to Geant4 uncertainty quantification and performance improvement

M Batič<sup>1,2</sup>, M Begalli<sup>3</sup>, M Han<sup>4</sup>, S Hauf<sup>5</sup>, G Hoff<sup>1,6</sup>, C H Kim<sup>4</sup>,  
M Kuster<sup>7</sup>, M G Pia<sup>1</sup>, P Saracco<sup>1</sup>, H Seo<sup>4</sup>, G Weidenspointner<sup>8,9</sup>,  
A Zoglauer<sup>10</sup>

<sup>1</sup> INFN Sezione di Genova, Genova 16146, Italy

<sup>2</sup> Jozef Stefan Institute, 1000 Ljubljana, Slovenia

<sup>3</sup> UERJ, 20550-013, Rio de Janeiro, RJ, Brazil

<sup>4</sup> Hanyang University, Seoul 133-791, Korea

<sup>5</sup> Technische Universität Darmstadt, IKP, Germany

<sup>6</sup> Pontificia Universidade Católica do Rio Grande do Sul, Porto Alegre, Brazil.

<sup>7</sup> European XFEL GmbH, Hamburg, Germany

<sup>8</sup> Max-Planck-Institut für extraterrestrische Physik, 85740 Garching, Germany

<sup>9</sup> MPI Halbleiterlabor, 81739 München, Germany

<sup>10</sup> Space Sciences Laboratory, University of California at Berkeley, Berkeley, CA 94720, USA

E-mail: Maria.Grazia.Pia@cern.ch

**Abstract.** Ongoing investigations for the improvement of Geant4 accuracy and computational performance resulting by refactoring and reengineering parts of the code are discussed. Issues in refactoring that are specific to the domain of physics simulation are identified and their impact is elucidated. Preliminary quantitative results are reported.

## 1. Introduction

The Geant4 [1, 2] simulation toolkit is nowadays a mature software system: at the time of writing this paper, its reference publication [1] has collected more than 3000 citations [3]. The development of Geant4 started in 1994 as the RD44 [4] project and its first version was released at the end of 1998; since then, Geant4 has been used in a wide variety of experimental applications, while further code development continued, also motivated by new requirements originating from the experimental community.

Over the 18 years elapsed since the start of Geant4 development, the object oriented paradigm has evolved from the status of pioneering technology into established methods and software design techniques, while new compilers nowadays support features of the C++ language that were not practically available in earlier versions.

Methods and techniques have been developed over the years to cope effectively with the evolution of large scale object oriented systems by providing guidance for the improvement of the design of existing software; some of them are now well established components of the software engineering body of knowledge, and are documented in classical textbooks [5], [6], [7].

While the architectural design of Geant4 established in RD44 has demonstrated its soundness by supporting the growth of the toolkit and its applications in multidisciplinary environments, Geant4 could profit from exploiting established refactoring and reengineering techniques to improve the design in some parts of the code, especially those that have been subject to extensive evolution in recent years, or that should accommodate new experimental requirements.

A project in progress investigates the benefits that could derive from the exploitation of these techniques, namely in Geant4 physics domain: this investigation is not limited to evaluating effects that are typically associated with design improvements, such as ease of maintenance and facilitation of further extensions of functionality, but it also estimates their impact on the quantification of Geant4 simulation accuracy and its computational performance. This project also explores issues, and methods to address them, that, while conceptually similar to those encountered in conventional refactoring projects, are specific to the environment of a large scale physics software system such as Geant4.

This conference paper summarizes the main ideas underlying the ongoing R&D (research and development) pursued by the authors, and a few initial results of the activity in progress; extensive details are meant to be documented in dedicated publications in scholarly journals.

## **2. Vision**

The activities documented in this paper - refactoring and reengineering parts of Geant4 code - are carried out within the scope of a wider scientific research vision, focused on the investigation of fundamental topics in particle transport. Research is articulated over two main areas, that are logically and technically intertwined: the assessment of the state-of-the-art in physics modeling for particle transport (with consequent improvement of Geant4 physics to reflect the state-of-the-art, if not yet achieved), and the objective quantification of the uncertainty of simulation results.

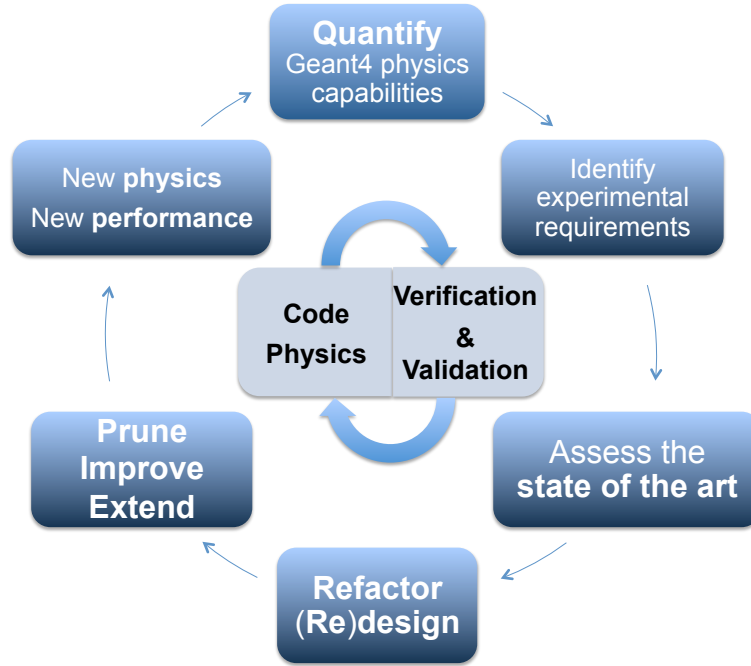
Refactoring and reengineering techniques support this scientific vision by contributing to improve the software design, hence the transparency of Geant4 physics modeling, the capability of quantifying its accuracy at a fine grained level of detail, and the agility towards implementing state-of-the-art physics in a computationally effective environment.

The project adopts an iterative-incremental life-cycle model [8], which is illustrated in figure 1: while it is supported by a broad scientific vision, concrete deliverables are produced in the course of the activity, which are practically usable in the current simulation environment and respond to existing experimental issues.

## **3. Technical matters**

Refactoring and reengineering techniques are extensively documented in several books (e.g. [5], [6], [7]) and journal articles; the reader is referred to them for detailed information. The classical definitions of the two terms, as given in the above cited books, is reported in table 3 for convenience.

A recurrent term in the context of refactoring, also mentioned in the following sections, is “smell”; it was introduced in [5]. Code smell is a surface indication that there might be a deeper problem in the software system; by definition a smell is quick to spot (e.g. a long method). Nevertheless code smells do not always indicate a real problem; usually they are not bugs, i.e. they do not prevent the program from functioning correctly, rather they indicate weaknesses in design that may hinder further development or increase the risk of errors in the future.



**Figure 1.** Simplified illustration of the life-cycle model adopted in the investigation of the state-of-the-art in physics models for particle transport.

**Table 1.** Definition of relevant concepts.

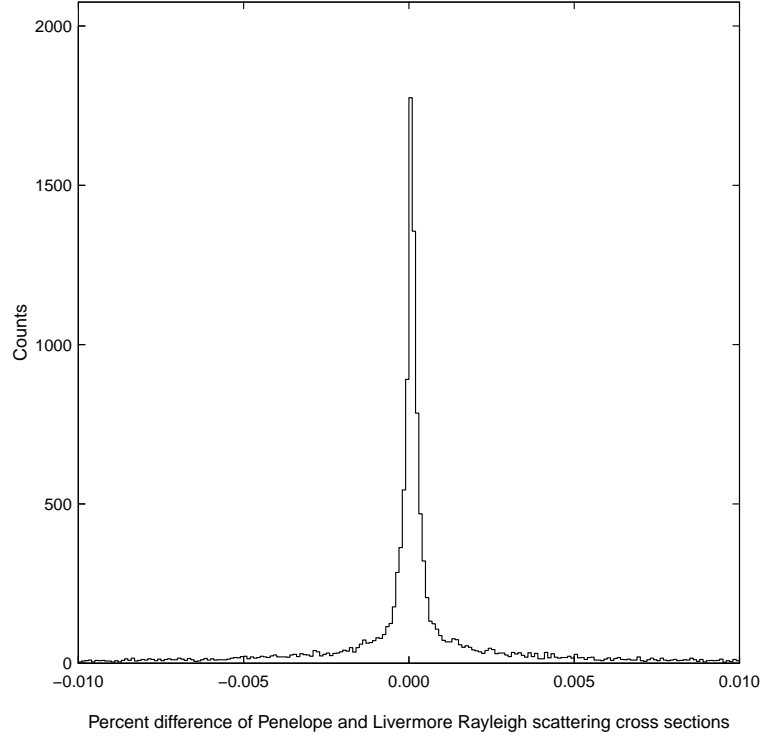
Concept	Definition	Source
Refactoring	“Refactoring is the process of changing a software system in such a way that it does not alter the external behavior of the code yet improves its internal structure”	[5]
Reengineering	Reengineering “seeks to transform a legacy system into the system you would have built if you had the luxury of hindsight and could have known all the new requirements that you know today”.	[7]

#### 4. R&D in Geant4 electromagnetic physics

Geant4 electromagnetic physics package has been subject to major evolution since the first release of Geant4: it has included developments for new functionality, and has been the playground for extensive design modifications. Several “code smells” listed in Fowler’s seminal “Refactoring” book can be identified in this package (e.g. long methods, long parameter lists etc.); standard refactoring techniques can be applied to attempt to improve the quality of the software design.

Neither these symptoms nor the techniques to deal with them will be analyzed in detail here; rather the attention is focused on less conventional “smells” identified in this code, which are specific to the physics simulation environment, although conceptually similar to typical problems addressed by refactoring techniques.

Duplicated code is “number one in the stink parade” according to [5]. In the context of Geant4 electromagnetic physics in-depth analysis of the code and its physics performance has also identified duplicated physics functionality and duplicated atomic parameters, which appear as different code.



**Figure 2.** Percent difference of Rayleigh scattering cross sections calculated by Geant4 “Penelope” and “Livermore” models.

Duplicated physics models are models that provide identical functionality and simulation accuracy in different implementations. An example is the identical simulation of Rayleigh scattering in the Geant4 models identified as “Livermore” and “Penelope”, respectively associated with different classes: both models, as they are released in Geant4 9.5, are based on the interpolation of cross sections and form factors tabulated in the EPDL97 [9] data library. Figure 2 shows the percent difference between the cross sections calculated by the two models: the very small differences appearing in the histogram are the results of EPDL97 data interpolation in the respective implementations. The two implementations exhibit identical compatibility with experimental data, quantified by means of goodness-of-fit-tests [10, 11]. Further details on this issue can be found in [12]. This duplication of functionality is the result of recent evolutions in the Geant4 implementation of Penelope-like models: the Penelope-like Rayleigh scattering implemented in the first reengineering of Penelope [13] was based on the original Penelope model, which did not use EPDL97; the current Geant4 implementation was reengineered from a later version of Penelope (Penelope 2008), where the original Rayleigh scattering model had been replaced by one based on EPDL97.

Duplicated physical parameters were also identified in Geant4 electromagnetic package: for instance, different sets of atomic electron binding energies. The duplication of atomic data in different parts of the code is prone to generate inconsistencies in simulation observables depending on them. Significant investment in software redesign is necessary to deal with atomic data consistently, while ensuring optimal accuracy for all the simulation models that use them; further details are discussed in [14].

Automated techniques for the identification of duplicated code are available to facilitate the refactoring process [7]; nevertheless, to a large extent the identification of “bad smells” in the

code relies on the intuition of experienced software developers. The duplicated physics models and parameters in Geant4 electromagnetic package discussed above were identified thanks to rigorous physics validation analyses, complemented by in-depth code review.

Duplicated code and duplicated physics functionality implemented in Geant4 should be pruned. Also code that exhibits inferior physics functionality than the model identified as the state-of-the-art, and comparable or inferior computational performance should be pruned.

An undesirable physics feature identified in the current design of Geant4 electromagnetic physics is the coupling between total cross section and final state calculation in the same class. Greater flexibility in choosing the two modeling approach independently would ensure the optimization of physics configuration in experimental scenarios that are especially concerned with simulation accuracy or computational performance. It is worthwhile to note that the responsibilities for cross section calculation and final state generation have been decoupled in Geant4 hadronic physics domain since the RD44 phase.

Another undesirable feature of the current design of Geant4 electromagnetic package is due to dependencies on other parts of the software: for instance, a full scale Geant4-based simulation application, involving a geometry model, is required even for testing low level physics modeling, such as a cross section calculation.

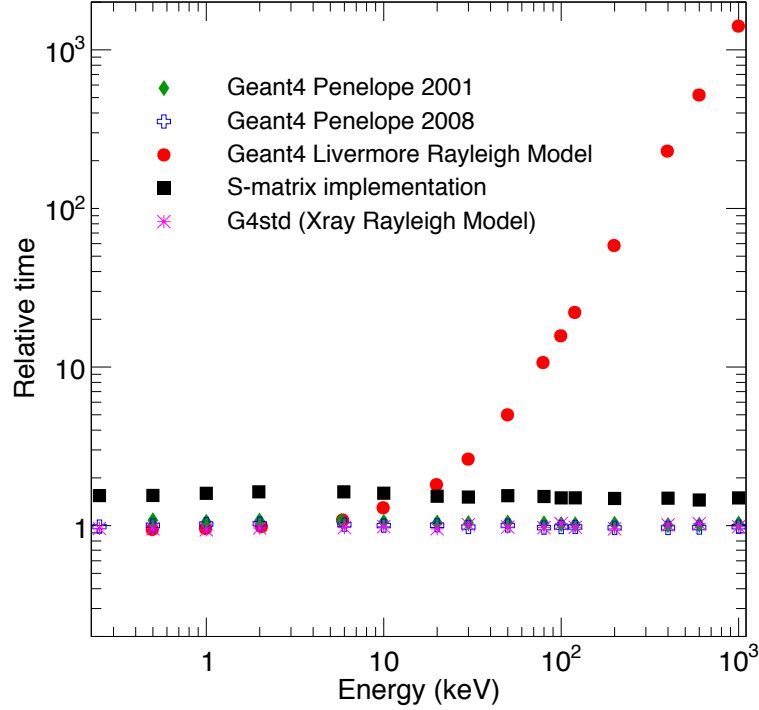
The last two features are associated with an inadequate problem domain analysis: refactoring techniques are not sufficient to deal with these deficiencies, which require improving the problem domain decomposition to provide sound foundation for the software design. A prototype design that addresses these issues, deriving from more effective problem domain analysis, was presented at a previous conference [15]; this design approach, which exploits generic programming techniques, has been adopted in a recent large scale study of photon elastic scattering simulation [12], where it demonstrated its ability to support the development of a large variety of physics models and has enabled in-depth verification and validation of their capabilities.

A new model of photon elastic scattering based on S-matrix calculations (SM) has been developed in the course of this study, which improves the compatibility with experiment by approximately a factor two with respect to models currently implemented in Geant4 (EPDL97), although at the price of some deterioration of computational performance [12]. Alternatively, more modern form factor calculations (MFASF) can improve the compatibility with experimental data by approximately 40% with respect to current Geant4 models without additional computational burden. The main results of the experimental validation process are summarized in table 2; the efficiency reported in the table represents the fraction of test cases where the  $\chi^2$  test finds a model compatible with experimental data with 0.01 significance. The full set of results is documented in [12].

**Table 2.** Efficiency of photon elastic scattering models at reproducing experimental data

Scattering angle	EPDL97	SM	MFASF
$0^\circ \leq \theta \leq 180^\circ$	$0.38 \pm 0.06$	$0.77 \pm 0.05$	$0.52 \pm 0.06$
$\theta \leq 90^\circ$	$0.40 \pm 0.06$	$0.82 \pm 0.05$	$0.54 \pm 0.06$
$\theta > 90^\circ$	$0.06 \pm 0.06$	$0.59 \pm 0.05$	$0.12 \pm 0.06$

The refactoring process requires a sound and fine-grained testing system to ensure that it does not modify the functionality of the code. In the project in progress it is complemented by thorough testing for the validation of Geant4 physics models and the evaluation of their computational performance. This fine-grained testing process allows the identification of the various elements that contribute to the functionality of a class, and the quantification of their accuracy and computational performance, to a great level of detail. One of the results of this process is the quantification of the contributions to computational performance intrinsically



**Figure 3.** Computational performance of various Rayleigh scattering models “Livermore” models.

due to physics modeling, and those related to other algorithms. For instance, it has allowed the identification of an inefficient sampling algorithm as the responsible for the apparently poor computational performance of Rayleigh scattering implementation in the so-called Geant4 “Livermore model” shown in figure 3. Once the inefficient sampling algorithm is replaced by a more efficient one, the computational performance of that physics drops significantly, becoming equivalent to the performance of the fastest models shown in figure 3.

A further attempt to improve computational performance in Geant4 physics domain by shifting the emphasis from algorithms to data libraries is currently the object of exploration: it consists of minimizing the use of algorithms in physics modeling, while privileging the use of data libraries. A related study in progress evaluates the possibility of merging physics models providing functionality for different energy rangess by smoothing data tabulations derived from them, rather than through algorithms as it is currently done in Geant4 [16]. If the ongoing prototype investigations prove that these methods would achieve significant performance improvements without degradation of physics accuracy, a more extensive reengineering process will be justified. Preliminary investigations of reengineering Geant4 data management domain [17] have demonstrated significant gains in computational performance.

## 5. R&D in Geant4 radioactive decay simulation

Significant effort has been invested into assessing the accuracy of Geant4 radioactive decay simulation, and improving its physics accuracy and computational performance.

The reengineering process has improved the design, which is now based on a sound domain decomposition and is characterized by well identified responsibilities. A class diagram illustrating the main features of the reengineered software design is shown in figure 4.

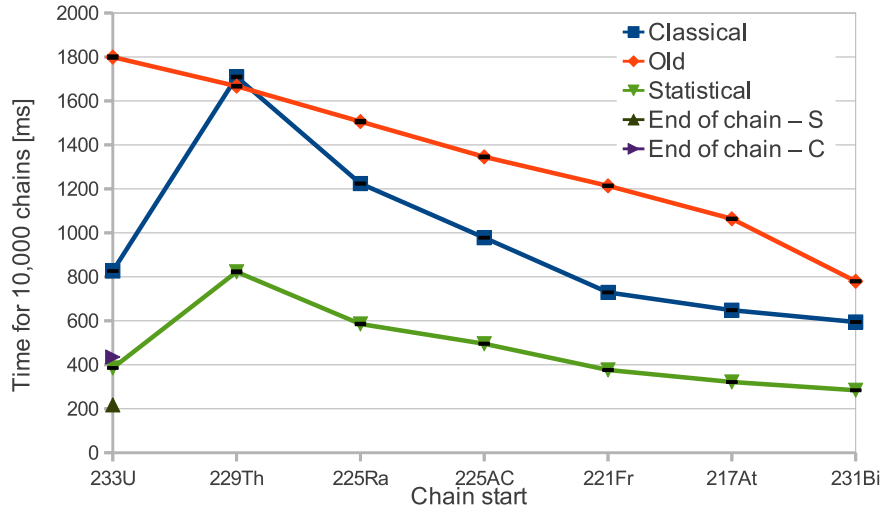
Refactoring results in improved computational performance, as can be observed in figure



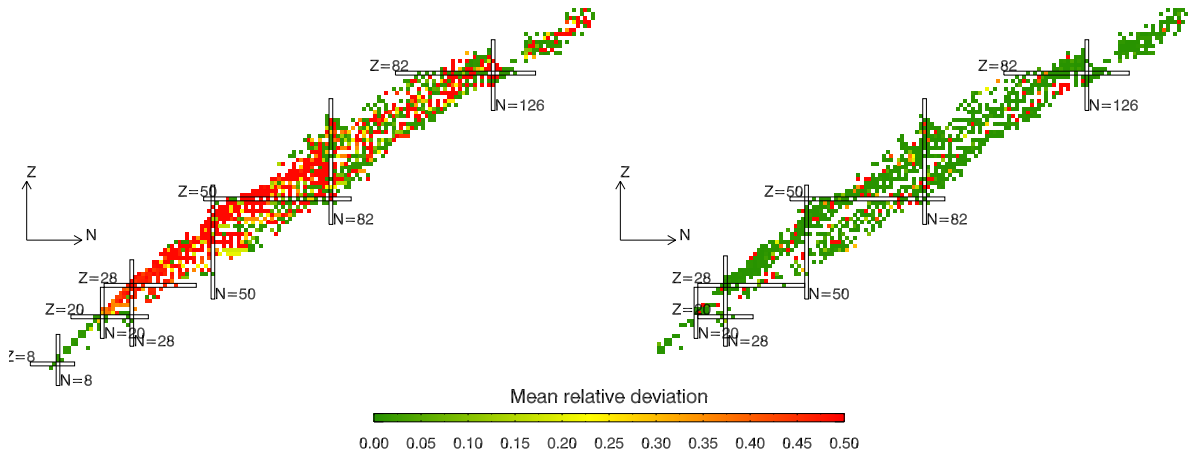
**Figure 4.** Class diagram of Geant4 radioactive decay resulting from the reengineering process.

5. Further improvement in computational performance are achieved by a new model, which is based on a different conceptual approach with respect to the model currently implemented in Geant4, since it treats the decay chain in statistical terms. The reengineered software design can accommodate both modeling alternatives, whose different underlying approaches may address different use cases: this extension of functionality would have not been possible in the context of the original package design.

Extensive details of the reengineering process and its achievements are documented in [18], along with the results of the experimental validation of the software.



**Figure 5.** Computational performance of radioactive decay simulation when decaying  $^{233}\text{U}$ : current Geant4 code (red), reengineered code (blue) and new algorithm (green).



**Figure 6.** Nuclide charts showing the median relative intensity deviations from reference data per isotope for Auger electron emission for the original Geant4 radioactive decay code (left), and for the reengineered one (right), which exploits improved atomic parameters.

The reengineering of Geant4 radioactive decay code profits from the improvements to atomic parameters discussed in section 4 regarding the refactoring of Geant4 electromagnetic physics, resulting in better agreement with respect to reference data: an example of results is illustrated in figure 6, where nuclide charts show the median relative intensity deviations from reference data per isotope for Auger electron emission for the original Geant4 radioactive decay code, and for the reengineered one, which exploits improved atomic parameters.

## Conclusion

Ongoing activities concerning the improvement of Geant4 design by means of refactoring and reengineering techniques have achieved significant results, that demonstrate their contribution



to improved physics accuracy and computational performance.

Further evaluations are in progress.

## Acknowledgments

The authors are grateful to the CERN Library for the support provided to this study.

## References

- [1] Agostinelli S *et al* 2003 *Nucl. Instrum. Meth. A* **506** 250
- [2] Allison J *et al* 2006 *IEEE Trans. Nucl. Sci.* **53** 270
- [3] <http://apps.webofknowledge.com>
- [4] Giani S *et al* 1998 *CERN-LHCC-98-044*
- [5] Fowler M 1999 *Refactoring: Improving the Design of Existing Code* Addison Wesley
- [6] Feathers M C 2004 *Working Effectively with Legacy Code* Prentice Hall
- [7] Demeyer S *et al* 2003 *Object-oriented reengineering patterns* Morgan Kaufmann
- [8] Jacobson I, Booch G and Rumbaugh J 1999 *The Unified Software Development Process* Addison-Wesley
- [9] Cullen D *et al* 1997 *EPDL97, the Evaluated Photon Data Library* UCRL-50400, Vol. 6, Rev. 5
- [10] Cirrone G A P *et al* 2004 *IEEE Trans. Nucl. Sci.* **51** 2056
- [11] Mascialino B *et al* 2006 *IEEE Trans. Nucl. Sci.* **53** 3834
- [12] Batic M *et al* 2012 *IEEE Trans. Nucl. Sci.* **59** 1636 <http://arxiv.org/abs/1206.0498>
- [13] J. Baro J *et al* 1995 *Nucl. Instrum. Meth. B* **100** 31
- [14] Pia M G *et al* 2011 *IEEE Trans. Nucl. Sci.* **58** 3246
- [15] Pia M G *et al* 2010 *J. Phys.: Conf. Ser.* **219** 042019
- [16] Batic M *et al* 2012 *Algorithms and parameters for improved accuracy in physics data libraries* these proceedings
- [17] Han M *et al* 2011 *J. Phys.: Conf. Ser.* **331** 042010
- [18] Hauf S 2012 *Background simulations for the IXO and ATHENA Wide Field Imager and the Development of a new Radioactive Decay Code for Geant4*, PhD Thesis, Tech. Univ. Darmstadt